To Propose a Reviewer Dispatching Algorithm for Networked Peer Assessment System

Eric Zhi-Feng Liu, Sunny San-Ju Lin, Shyan-Ming Yuan
National Chiao Tung University
E-Mail: totem@cis.nctu.edu.tw

Abstract
Despite their increasing availability on the Internet, networked peer assessment systems lack feasible automatic dispatching algorithm of student's assignments and ultimately inhibit the effectiveness of peer assessment. Therefore, this study presents a reviewer dispatching algorithm capable of supporting networked peer assessment system in order to automatically dispatch student's assignments to their peer reviewers.

Reviewer Dispatching Algorithm
We set four constraints for dispatching reviewers. The first is that author should not review his (or her) own assignment. The second is that how many reviewers per assignment should set up via instructor. The third is that how many reviewers per assignment should equal to how many assignments reviewed by each student. The forth is that the reviewers of first assignment are fixed under k rounds but the reviewers of second or another assignment should be different to maintain the fairness.

Then, we would like to model these constraints via constraint satisfaction problem. In order to illustrate this process in an easy way, we hypothesized that when there are three students. In the first step, we write down a 3x2 matrix (Table 1).

<table>
<thead>
<tr>
<th>Col1</th>
<th>Col2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Student1</td>
<td>X1</td>
</tr>
<tr>
<td>Student2</td>
<td>X3</td>
</tr>
<tr>
<td>Student3</td>
<td>X5</td>
</tr>
</tbody>
</table>

Table 1: A 3x2 matrix for modeling the constraints of three students
In the second step, we defined the X_i = {1, 2, or 3}, <> should not transitive but symmetric, and the following constraints (C1, C2, and C3):

C1: X1<>1, X1<>X2, X1<>X3, X1<>X5
C2: X3<>2, X3<>X4, X3<>X5
C3: X5<>3, X5<>X6

In third step, we utilized the recursive searching method for constraint satisfaction problem. The answers for this problem could be found in 3^6 (729) steps, also if there are four students then there are twelve variables and the execution steps could equal to 4^{12} (16,777,216). This result would make the dispatching task impossible, so we should seek another ways to solve this problem.

We listed the first possible solution for the constraint graph (left part of Fig. 1), but it is hard to find the regularity. Then, we listed the second possible solution for the constraint graph (right part of Fig. 1), and via observation I concluded a simple formula, \{X_i=3 if ((col# + 1) mod 3)=0, X_i=((col# + 1) mod 3)<>0\}, for calculating one of the answers of this problem. Surprisingly, dispatching three reviewers using this formula needs only twelve steps and saves 98% execution steps when compared with recursive searching method.
To standardize this algorithm, we have written down a C language like pseudocode (Table 2). The time complexity of this algorithm should be $O(n)$, $T(n) = 1 + 2n \leq c_1n = O(n)$ for some constant $c_1 > 2$, in the best case, $O(n^2)$, $T(n) = n - 1 + 2(n-1)n = 2n^2 - n - 1 \leq c_1n^2 = O(n^2)$ for some constant $c_1 \geq 2$, in the worst case, and $O(n^2)$, in the average case. The space complexity of this algorithm should be $O(1)$ to $O(n)$, if the program only save some important data (e.g. total number of students and some row numbers), in the best case, $O(n^2)$, $S(n) = k + n(n-1) = n^2 - n + k \leq c_1n^2 = O(n^2)$ for some constant $c_1 \geq 1$, in the worst case, and $O(n^2)$ in the average case. About k what value instructors should set, we suggested set k between three and six, and not set k too large (e.g. > 20) to reduce the resistance from students.

/* n is the total number of students */ /* k is the number of reviewers */ /* p is 1 by k array */ /* s is a n by k array */

Input: n, k, p, s, Result: s
Assessor_Dispatching (n, k,p, s)
1 Choose k different number between 1 to n-1 to p array
2 For i = 1 to k
3 do w = p[i]
4 for j = 1 to n
5 do $s[j,w] = (w + 1 \mod n)$, w = w + 1
6 return s

Table 2: The pseudocode of dispatching algorithm.

Acknowledgments

The authors would like to thank the National Science Council of the Republic of China under Contract Nos. NSC 90-2520-S-009-003 and NSC90-2520-S-009-002.

References